래치구조의 저면적 유한체 승산기 설계

Design of a Small-Area Finite-Field Multiplier with only Latches

李 光 燁*
Kwang-Youb Lee*

요 약

본 논문은 암호화 장치 및 오류정정부호화 장치 등에서 핵심적으로 사용되고 있는 유한체승산기(finite-field multiplier)의 최적화된 구조를 제안한다. 제안된 구조는 LFSR(Linear Feedback Shift Register)구조를 갖는 유한체 승산기에서 소비전력과 회로면적을 최소화 하여 기존의 LFSR 구조를 바탕으로 하는 유한체 승산기에 비하여 효율적인 승산을 이루도록 한다.

기존의 LFSR 구조의 유한체 승산기는 m비트의 다항식을 승산 하는데 $3 \cdot m$개의 플립플롭(flip-flop)이 필요하다. 1개의 플립플롭은 2개의 래치(latch)로 구성되기 때문에 $6 \cdot m$개의 래치가 소요된다. 본 논문에서는 $4 \cdot m$개의 래치(m 개의 플립플롭과 $2 \cdot m$개의 래치)로 m 비트의 다항식을 승산 할 수 있는 유한체 승산기를 제안하였다. 본 논문의 유한체 승산기는 기존의 LFSR 구조의 유한체 승산기에 비하여 회로구현에 필요한 래치의 개수가 1/3(약 33%)이 감소하였다. 결과적으로 기존의 방법에 비하여 저 소비전력 및 저 면적의 유한체 승산기를 암호화 장치 및 오류정정부호화 장치 등에서 효과적으로 사용이 가능하다.

Abstract

An optimized finite-field multiplier is proposed for encryption and error correction devices. It is based on a modified Linear Feedback Shift Register (LFSR) which has lower power consumption and smaller area than prior LFSR-based finite-field multipliers. The proposed finite field multiplier for GF(2n) multiplies two n-bit polynomials using polynomial basis to produce z(x)=a(x)*b(x) mod p(x), where p(x) is a irreducible polynomial for the Galois Field. The LFSR based on a serial multiplication structure has less complex circuits than array structures and hybrid structures. It is efficient to use the LFSR structure for systems with limited area and power consumption.

The prior finite-field multipliers need $3 \cdot m$ flip-flops for multiplication of m-bit polynomials. Consequently, they need $6 \cdot m$ latches because one flip-flop consists of two latches. The proposed finite-field multiplier requires only $4 \cdot m$ latches for m-bit multiplication, which results in 1/3 smaller area than the prior finite-field multipliers. As a result, it can be used effectively in encryption and error correction devices with low-power consumption and small area.

Key words : Finite-Field, Multiplier, Cryptography, LFSR, Smart card

*西京大學校 컴퓨터工學科
(Dept. of Computer Engineering, Seokyeong Univ.)

接受日:2002年 8月 16日, 修正完了日:2003年 7月 14日 ※본 논문에서는 IDEC(반도체설계교육센터)의 장비를 활용하였습니다.

Ⅰ. 서 론

최근 정보화 단말기가 급성장하면서 주목을 받고 있는 스마트카드와 공개키암호(public-key cryptography)는 20여년전 부터 연구되어 왔다. 그러나 IC 카드에 적합한 공개키암호 회로의 구현방법이 발달하지 못하여 스마트카드와 공개키암호가 결합된 것은 불과 수년전의 일이 되었다. 최근에는 RSA, US Digital Signature Strandard, DES와 같은 암호 알고리즘이 많은 IC 카드에 응용되고 있다.

스마트카드에 사용될 수 있는 공개키의 개념은 1976년 W. Diffie와 M. E. Hellman이 "New Directions in Cryptography"에서 공개키 암호의 개념을 처음 소개하였다. 이후 1978년 소인수분해의 어려움에 기반을 둔 RSA가 소개되어 지금까지 넓게 사용되고 있다. 그러나 RSA는 비도를 높이기 위해 1024 비트 이상으로 확장되는 추세로, 스마트카드와 같이 제한된 면적에 탑재되는데 어려움이 있다. 스마트카드에 내장되는 칩에는 마이크로프로세서, 메모리, 암호화프로세서, 입출력 제어회로 등이 포함된다. 그러나 스마트카드 칩은 카드 reader 시스템과의 인터페이스를 원할히 하기 위하여 그 규격이 국제적으로 정하여져 있으며 칩의 크기가 한정되어 있다. 따라서 마이크로프로세서, 메모리, 암호화프로세서, 입출력제어회로와 같은 기능이 한정된 공간에서 구현되어야 하기 때문에 최소면적의 회로구조를 갖추어야 한다. 1987년 Koblitz와 Miller는 공개키 암호화에 타원곡선(ECC) 알고리즘을 적용하였다. ECC(Elliptic Curve Cryptography)[1][2]는 적은 비트로 높은 비도를 보이기 때문에 최근 스마트 카드와 같은 IC카드[3]의 암호화구현에 사용되는 추세에 있다.

타원곡선 알고리즘을 적용한 암호프로세서는 유한체 승산기, 제산기, 가산기, 제곱기로 구성되며 제산기와 제곱기는 승산기로도 구현이 가능하기 때문에 승산기가 가장 핵심적인 역할을 한다. 유한체 승산기는 다양한 구현 방법을 갖지만 일반적으로 LFSR(Linear Feedback Shift Register) 구조의 직렬(serial) 승산 구조와 Systolic array 구조를 적용한 병렬(parallel) 승산 구조가 널리 사용된다. 시스톨릭 병렬 승산기는 2차원 배열(array) 유한체 승산기로 직렬 승산기에 비하여 m배의 하드웨어를 사용하여 m배의 계산 속도를 얻어내는 방식으로 가격 대 성능비에서 효율이 떨어진다.

반면, Bit-serial 구조의 승산기는 LFSR(Linear Feedback Shift Register)[4] 를 주로 사용하여 설계되는데 암호비트수에 비례하여 지연시간이 증가하지만 bit-parallel 보다 게이트의 수를 감소시키는 방법이다.이 가운데 LFSR구조는 승산을 수행하는 속도가 병렬승산에 비하여 느린 단점은 있지만 회로구현이 간편하고 적은크기의 회로를 요구하기 때문에 사용범위가 넓다. LFSR구조의 승산기는 Edoardo D. Mastrovito가 제안한 방법을 바탕으로 하여 여러가지로 개선되어 왔다[5][6]. 직렬 승산방식인 LFSR구조의 단점인 느린 수행속도를 개선하는 방법이 제안되었다. 이 방법은 t 배의 속도향상을 위하여 $t \cdot m$개의 레지스터가 추가되어 회로의 크기가 증가하였다[7]. 또한, 레지스터의 수를 증가시키지 않고 속도를 개선하는 방법도 제안되었다[8].

본 논문에서는 제한된 공간에서 최소면적의 암호화 프로세서 회로를 구현할 수 있도록 적은회로규모 및 저소비전력의 승산기 구조를 제안하였다. LFSR구조의 승산기 회로는 레지스터를 바탕으로 구성되기 때문에 회로크기와 소비전력을 결정하는 것은 레지스터이다. 승산 데이터의 길이가 m 비트일때 입력 데이터와 승산 결과 데이터를 저장하기 위하여 $3 \cdot m$개의 레지스터가 필요하다. 이때 레지스터는 쉬프트(shift) 기능을 갖기 때문에 클록의 에지(edge)에서 동기화되는 플립플롭으로 구현되어야 한다. 쉬프트 레지스터를 클록의 레벨(level)에서 동기화되는 래치(latch)로 구현하면 클록의 레벨 구간에서 데이터의 레이싱이 발생하여 쉬프트가 이루어지지 않는다.

플립플롭의 회로크기와 소비전력은 래치의 2배에 해당되기 때문에 LFSR구조의 승산기 회로를 저면적, 저소비전력으로 구현하기 위한 가장 효과적인 방법은 쉬프트 레지스터에서 플립플롭을 래치로 대치하는 방법을 사용하는 것이다.

본 논문에서는 LFSR구조의 유한체 승산기에서 쉬프트 레지스터를 구성하는 플립플롭을 래치로 대치하여 기존 회로에서 래치 개수의 1/3을 줄임으로써 전체 회로의 크기와 소비전력을 크게 줄이는 방법을 제안한다.

Ⅱ. LFSR 구조의 유한체 승산기

유한체 연산에서 연산식은 standard base, normal base, dual base 등으로 표현되는데 일반적으로 standard base 표현이 연산기 구현에 용이하기 때문에 널리 사용된다. Polynomial base표현은 Standard base 표현에 속하는 것이다. Polynomial base 유한체상에서 일반적인 승산기구조는 bit-serial 방식과 array 방식으로 구현된다. Array 방식의 경우 고속의 동작이 가능하지만 구현에 필요한 하드웨어면적이 크기 때문에 휴대형의 정보단말 장치에서는 bit-serial 구조를 채택하고 있다.

Bit-serial 구조의 승산기는 LFSR(Linear Feedback Shift Register)를 주로 사용하여 설계되는데 암호비트수에 비례하여 지연시간이 증가하지만 Array 방식 보다 게이트의 수를 감소시키는 방법이다.

2-1. Conventional LFSR 승산기

직렬 유한체 승산의 표현식은 다음 식(1)과 같이 표현된다.

$$Z = A \cdot B = \sum_{i=0}^{m-1} b_i(A\alpha_i)$$
$$= [\cdots[[b_0(A) + b_1(A\alpha)] + b_2(A\alpha^2)] + \cdots]$$
$$+ b_{m-1}(A\alpha^{m-1}) \quad (1)$$

식(1) 을 회로로 구현하면 그림 1과 같은 bit-serial 승산기를 얻을 수 있다. 그림 1은 일반적으로 LFSR(Linear Feedback Shift Register) 구조라고 부른다.

그림 1은 m비트의 입력 a, b를 승산하고 그 결과를 기약다항식 p로 나누어(modulo) m비트의 출력 z를 생성하는 승산기 회로를 나타낸다. 대표도에서 a0 부터 am-1 까지 m개의 레지스터, b0 부터 bm-1 까지 m개의 레지스터는 승산기의 입력 데이터를 저장하는 레지스터를 나타내며 z0 부터 zm-1 까지 m 개의 레지스터는 승산 결과를 저장하는 레지스터를 나타낸다. 또한, p0부터 pm-1까지 m개 비트는 기약다항식의 계수(coefficient) 를 나타낸다. 기약 다항식의 계수는 궤환(feedback)을 의미하는 것으로 계수값이 1인 곳에서 궤환이 발생하여 a레지스터 값과 덧셈이 이루어진다. 그림 1에서 ⊕는 덧셈을 의미하며 XOR게이트로 구현된다. 기약다항식의 계수는 configurable 승산기 구조에서는 레지스터에 저장하여 다양한 형식의 다항식을 사용할 수 있도록 하고 있으나 본 논문의 설명에서는 기약다항식의 계수는 고정된(hard-wired) 것으로 간주한다.

그림 1 에서 Z 레지스터는 제1입력값과 제2입력값을 AND 한 결과와 자신의 값을 덧셈하여 다시 저장하는 역할을 하며 제1입력값과 제2입력값이 m번의 쉬프트를 수행한 후에 남아있는 값이 최종 승산결과값이 된다.

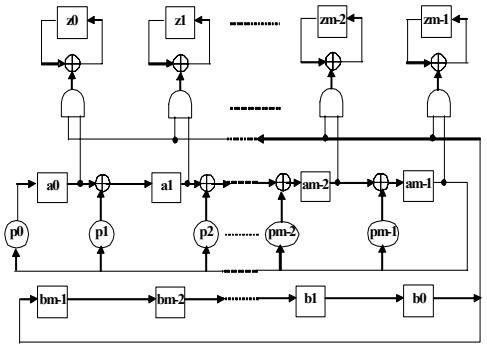


그림1. LFSR구조를 이용한 직렬유한체승산기
Fig. 1 A serial finite-field multiplier using LFSR architecture

2-2. 2배속 직렬유한체 승산기

기본적인 LFSR구조의 직렬 유한체 승산기의 처리속도를 개선하기 위하여 여러 가지 연구가 진행되었고 최근에는 그림 2와 같은 구조가 발표되었다[7]. 이 구조는 식(2)를 2부분으로 나누어 2배의 속도향상을 얻는 방법으로 식(1)를 식(2)과 같이 계수가 짝수인 부분과 홀수인 부분으로 나눈다. 식(3)의 표현을 이용하면 x^2의reduction으로Zeven(x)와 Zodd(x)를 동시에 수행하여 throughput을 2배 증가시킬 수 있다. 이 방법을 이용하여 회로를 구현하면 그림 2와 같이 Zeven(x)회로에는 m개의 AND게이트, m개의 XOR게이트, m개의 짝수 차수에 대한 결과 레지스터가 있고, 이와 비슷하게 Zodd(x)회로에는 m개의 AND게이트, m개의 XOR게이트, m개의 홀수 차수에 대한 결과 레지스터가 존재한다. 따라서, 그림 2의 방법으로 기본적인 LFSR 승산기보다 속도를

2배 개선하면 m개의 레지스터가 더 소요되는 단점이 있다.

$$Z_{even}(x) = [b_0A(x) + \ldots + b_{m-3}x^{m-3}A(x) + b_{m-1}A(x)] \bmod P(x)$$

$$\begin{aligned} Z_{odd}(x) &= [b_1xA(x) + b_3x^3A(x) + \ldots + b_{m-2}x^{m-2}A(x)] \bmod P(x) \\ &= x[b_1A(x) + b_3x^2A(x) + \ldots + b_{m-2}x^{m-3}A(x)] \bmod P(x) \end{aligned} \quad (2)$$

$$\begin{aligned} x^2A(x) &= a_0x^2 + a_1x^3 + a_2x^4 + \ldots + a_{m-2}x^m + a_{m-1}x^{m+1} \\ &= a_{m-2}p_0 + (a_{m-2}p_1 + a_{m-1}p_0)x + (a_{m-2}p_2 + a_{m-1}p_0 + a_0)x^2 + \ldots + (a_{m-2}p_{m-1} + a_{m-1}p_{m-2} + a_{m-3})x^{m-1} \end{aligned} \quad (3)$$

2-3. 저면적 2배속 승산기

기존의 개선된 2배속 승산기 구조에서는 유한체 승산기의 계산 속도를 증가시키기 위해서 레지스터의 수를 승산 데이터의 비트 수만큼 증가시켜야 한다. 또한, exclusive 게이트의 수가 증가하고 연결선이 복잡해져 회로의 크기를 크게 증가시키게 된다.

회로의 크기를 증가시키지 않는 방법으로 새롭게 제안된 유한체 승산기의 구조는 그림 3에 수록한 것과 같이 레지스터나 exclusive 게이트의 수를 크게 증가시키지 않고 그림 1에 나타낸 종래의 유한체 승산기 보다 계산 속도를 2배 증가 시킬 수 있다[8].

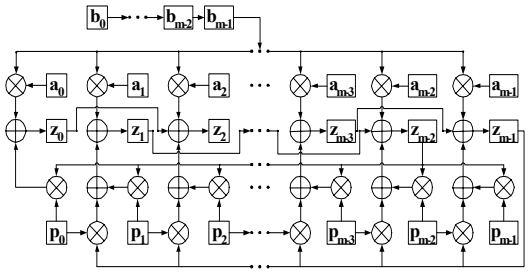


그림 2. 개선된 2배속 승산기의 Z_{even} 모듈의 회로도
Fig. 2 Z_{even} module of the improved 2-times faster multiplier

그림 2의 방법은 그림 1의 기본적인 형태의 유한체 승산기와 비교하여 승산속도를 2배 증가하기위해 승산 결과를 저장하는 Z 레지스터가 2배 크기로 증가하고 exclusive OR 게이트가 2배 증가하며 이를 연결하는 연결선의 복잡도가 크게 증가한다.

그러나 그림 3에 있는 논문의 유한체 승산기는 그림 1의 기본적인 형태의 유한체 승산기와 비교하여 레지스터 수에 변화가 없고 단지 궤환(feedback)이 발생하는 지점에서 레지스터가 1개 씩만 추가된다. 또한, exclusive 게이트 수에 변화가 없고 단지 AND 게이트 수가 2배로 증가한다.

그림 2는 그림 1에 비하여 승산속도를 2배로 증가하였으나 유한체 승산기 회로는 $3 \cdot m$개의 레지스터와 m개의 AND게이트, m+k개의 XOR게이트(k : 기약다항식 계수가 1인 개수. 즉, 궤환이 발생한 비트의 개수)로 구성되기 때문에 그림 1과 같은 크기의 회로를 갖는다. 그러나, 유한체 승산기 회로에서 레지스터가 전체 면적과 소비전력을 60% 이상 차지하기 때문에 저면적의 직렬 유한체 승산기를 설계하기 위해서는 레지스터구조를 개선하는 것이 필요하다.

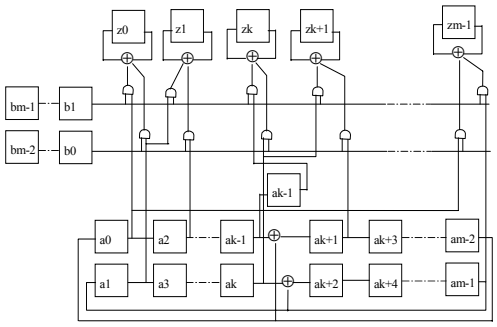


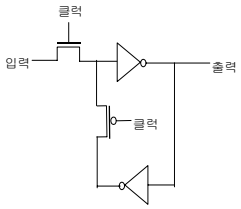
그림 3. 저면적 2배속 승산기 구조
Fig. 3 Architecture of the low area 2-times faster multiplier

Ⅲ. 저면적 직렬 유한체 승산기 설계

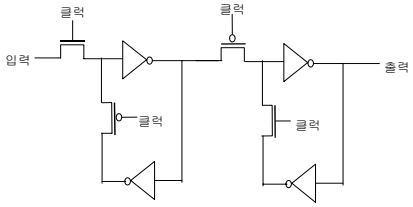
3-1. 플립플롭과 래치

일반적으로 레지스터는 플립플롭으로 만들어진다. 그림 4(a)는 일반적인 플립플롭의 회로구조를 나타낸다. 플립플롭은 클록의 에지(edge)에서 입력값을 출력으로 전달하기

때문에 내부적으로 두개의 래치가 연결되어 있는 구조를 갖는다. 그림 4(b)는 래치의 회로구조를 나타낸다. 결과적으로 플립플롭은 래치에 비하여 2배의 면적과 소비전력을 소모하게 된다. 따라서 유한체 승산기에 레지스터를 플립플롭 대신 래치를 적용하면 전체 회로의 크기와 소비전력을 상당히 개선할 수 있다. 그러나 래치는 클록의 레벨을 이용하여 입력 값을 출력에 전달하기 때문에 쉬프트레지스터를 래치로 구성할 경우 데이터의 lacing이 발생하여 플립플롭과 같이 매 클록마다 한 비트씩 이동시키는 기능을 수행할 수 없어 쉬프트레지스터에 적용하기가 어렵다.



(b) 래치
(b) Latch



(a) 플립플롭
(a) Flip-Flop

그림 4. 플립플롭과 래치 회로
Fig.4 Flip-Flop and Latch Circuits

3-2. 래치를 이용한 유한체 직렬 승산기

본 논문에서는 래치의 레이싱 문제를 클록 제어방법과 데이터 쉬프트(shift)순서 제어방법을 개선하여 LFSR 구조의 유한체 승산기를 플립플롭을 사용하지 않고 래치만으로 구현하는 방법을 제안한다.

그림 5는 기존의 유한체 승산기에서 제1입력값과 제2입력값을 저장하는 쉬프트레지스터를 플립플롭으로 구현한 일례를 나타낸다. 한 비트당 한 개의 플립플롭으로 구성되며 주클록에 따라 모두 동시에 동기화(syncronization)되어 쉬프트 동작이 이루어진다.

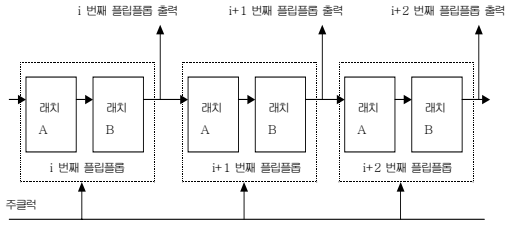


그림 5. 플립플롭을 이용한 쉬프트레지스터 구조
Fig. 5 Shift register architecture using flip-flops

그림 6은 본 논문에서 제안하는 쉬프트레지스터의 구조로 제1입력값과 제2입력값을 저장하는 쉬프트레지스터를 래치만으로 회로를 구현하였다. 제안된 방법에서 한 비트에는 래치 1개를 할당하고, 다음 비트에는 래치 2개를 할당한다. 그림 6에서 k번째와 k+2번째 비트에는 1개의 래치를 할당하였고 k+1번째 비트에는 2개의 래치를 할당하였다.

이와 같이 한 비트씩 건너서 1개의 래치로 구성된 레지스터와 2개의 래치로 구성된 레지스터를 교대로 연결하여 전체 쉬프트레지스터를 만든다.

그림 6의 제안된 구조에서 쉬프트레지스터의 실행방법을 설명하면 다음과 같다. 1비트의 데이터를 오른쪽으로 전달하기위해서는 다음과 같이 세가지 과정을 거쳐야 한다. 첫번째로 래치 2개로 구성된 레지스터에서 왼쪽 래치의 데이터를 오른쪽 래치에 전달한다(1). 이 전달과정에 필요한 클록을 클록 1 이라고 한다. 두번째로 래치 1개로 구성된 레지스터에서 오른쪽의 래치 2개로 구성된 레지스터에 데이터를 전달한다(2). 이 전달과정에 필요한 클록을 클록 2 라고 한다. 세번째로 래치 2개로 구성된 레지스터에서 오른쪽의 래치 1개로 구성된 레지스터에 데이터를 전달한다(3). 이 전달과정에 필요한 클록을 클록 3이라고 한다.

그림 7은 그림 6을 실행하기 위해 필요한 클록을 나타낸다. 본 논문에서 제안된 방법을 수행하기 위해서는 세 가지의 클록이 필요하면 각각 클록1, 클록2, 클록3 이라고 한다. 세 가지 클록은 주 클록(main clock)의 1주기(1 cycle)를 이용하여 그림 7과 같이 각각 생성된다.

이때, 클록1, 2, 3는 non-overlapping 이 되어야 하지만 클록 사이에 set-up time을 많이 두지 않아도 제안된 쉬프트 레지스터의 동작에는 레이싱을 발생하지 않는다. 또한, 스마트카드 등에서는 클록 주파수가 수십 MHz를 이용하기 때문에 충분하게 클록1,2,3을 발생할 수 있다.

본 논문에서 제안된 래치구조는 그림 1의 제1입력 레지스터 와 제2입력 레지스터에 모두 사용할 수 있으며 또한, 그림 1에서 승산결과를 저장하는 출력 레지스터에도 적용된다. 기존에는 출력 레지스터를 그림 5와 같이 래치 2개로 구성되어 있는 플립플롭으로 구현되었으며 클록의 에지에서 데이터의 저장이 이루어졌다. 그러나 본 논문에서는 그림 8에서와 같이 클록의 레벨을 이용하는 래치 한 개로 출력 레지스터를 구현하였다. 그림 8의 래치는 클록으로 그림 7의 클록 1을 사용하였다. 클록 3에서 제1입력 레지스터와 제2입력 레지스터가 1비트 쉬프트동작을 완료한다.

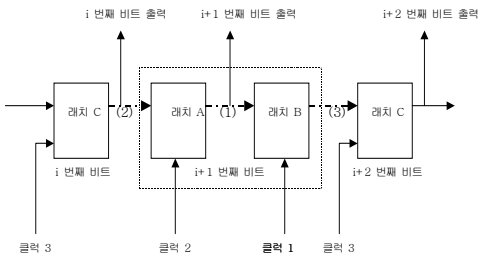


그림 6. 제안된 쉬프트레지스터의 구조 및 동작
Fig. 6 Proposed shift register architecture and operation

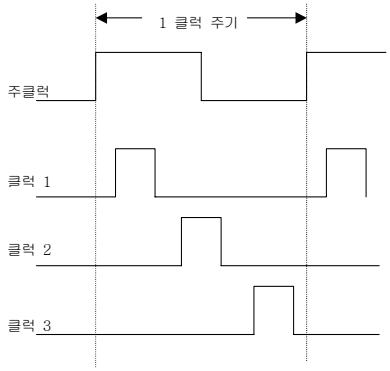


그림 7. 제안된 클록 구조
Fig. 7 Proposed clock architecture

따라서 다음 주기(cycle)의 클록 1이 시작할 때는 그림 1에서 AND 게이트의 출력이 안정한 상태(stable state)에 있기 때문에 클록 1에서 출력 레지스터가 자신의 값과 AND 게이트의 출력 값을 안정한 상태에서 덧셈동작을 수행할 수 있다.

본 논문에서 제안된 방법을 이용하여 래치만으로 LFSR구조의 유한체 승산기를 구현할 경우 m비트 승산기에서 $4 \cdot m$개의 래치가 소요된다. 반면, 기존의 플립플롭을 사용하는 방법으로 구현하면 모두 $6 \cdot m$개의 래치가 소요된다. 결론적으로 본 논문의 LFSR구조의 유한체 승산기는 기존의 방식에 비하여 30% 이상의 면적과 소비전력에서 효율을 얻을 수 있다.

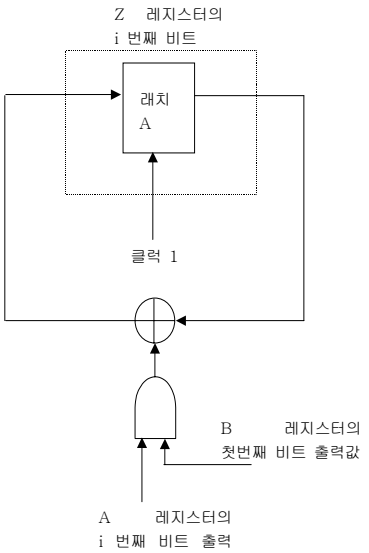


그림 8 래치를 이용한 출력 레지스터 구조
Fig. 8 Proposed output register architecture using latches

Ⅳ. 결 론

본 논문에서는 LFSR 구조를 개선하여 기존의 LFSR 구조 보다 면적과 소비전력 측면에서 효율적인 새로운 구조의 유한체 승산기를 제안한다. 기존의 LFSR 구조의 유한체 승산기는 플립플롭을 이용한 쉬프트레지스터를 기본으로 하여 설계하여 왔으나 본 논문에서는

래치만으로 쉬프트레지스터를 구성하여 전체 회로의 크기를 30% 이상 축소하였다. LFSR 구조의 유한체 승산기에서 레지스터가 60% 이상 차지하기 때문에 전체 LFSR 유한체 승산기의 회로 크기를 약 18%가량 축소할 수 있다. 타원곡선 암호프로세서와 같은 암호회로는 Galois Field상에서 승산, 제산, 가산, 제곱 등의 연산기로 구현이 되며 이 가운데 승산기가 가장 기본적인 연산기가 된다. 따라서 본 논문에서 제안된 저면적 유한체 승산기는 최근 스마트카드 등의 휴대형 정보단말 장치에서 널리 사용되고 있는 공개키 암호방식 가운데 타원곡선 암호알고리즘을 회로로 구현하는데 큰 장점을 갖는다.

참고문헌

[1] Michael Rosing,*Implementing Elliptic Curve Cryptography*, Manning Publications, 1999.

[2] G.B. Agnew, R.C. Mullin and S.A. Vanstone, "An Implementation of Elliptic Curve Cryptosystems Over F_2^{155}," *IEEE journal on selected areas in communications*, Vol 11. No. 5, June 1993.

[3] David Naccache David M'Raïhi, "Cryptographic Smart Cards," *IEEE MICRO*, Vol 16, No 3, pp. 14-23, June, 1996.

[4] H. Brunner, A. Curiger, and M. Hofstetter, "On Computing Multiplicative Inverses in GF(2^m)," *IEEE Transactions on Computers*, Vol. 42, No. 8, Aug. 1993.

[5] J.H Guo and C.L Wang,"Systolic Array Implementation of Euclid's Algorithm for Inversion and Division in GF(2^m)," *IEEE Transactions on Computers,* Vol. 47, No. 10, Oct. 1998.

[6] Edoardo D. Mastrovito, "VLSI Architecture for computations in Galois Fields," *Linkoping Studies in Science and Technology Dissertations* ,No. 242, 1991.

[7] Sangook Moon, J. Park, Y. Lee, "Fast VLSI architecture algorithms for high-security elliptic curve cryptographic applications," *IEEE Transactions on Consumer Electronics*, Vol, 47, No. 3, pp700-708, August 2001.

[8] 이광엽,김원종,장준영,배영환,조한진,"레지스터수의 증가 가 없는 고속 직렬 유한체 승산기," *한국통신학회 논문지*, Vol. 27, No. 10A, pp101-107, 10월 2002년.

저 자 소 개

李 光 燁(正會員)

1985년 서강대학교 전자공학과 학사. 1987년 연세대학원 전자공학과 석사. 1994년 연세대학원 전자공학과 박사 1989-1995년 현대전자 선임연구원 1995년부터 현재 까지 서경대학교 컴퓨터공학과 교수

주관심분야 : 마이크로 프로세서, 암호프로세서